(2)

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1261 October 1990

Exploiting the Redundancy of a Hand-Arm Robotic System

Claudio Melchiorri and J. Kenneth Salisbury

Abstract

In this report, a method for exploiting the redundant degrees of freedom of a hand-arm mechanical system for manipulation tasks is illustrated. The basic idea is to try to take advantage of the intrinsic capabilities of the arm and hand subsystems in terms of amplitude of motions, different velocity limits and degrees of precision for the achievement of a particular task. The Jacobian transpose technique, a well-known algorithm for the solution of the kinematic inverse problem, is at the core of the proposed method for the control of the hand-arm system. Different behaviors of the hand and of the arm are then obtained by means of constraints in $Null(\mathbf{J})$ added to the solution given by the Jacobian transpose method. The constraints are generated by non-orthogonal projection matrices, computed on the basis of the behavior desired from the system, without resorting to extended task space techniques. Comments about the computation of the constraints, and how to take advantage of them, are reported in the paper, as well as a description of the experimental activity currently in progress on a robotic system (a Puma 560 with the Salisbury Hand) at the Artificial Intelligence Laboratory, M.I.T.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for this research was provided by the University Research Initiative Program under Office of Naval Research contract N00014-86-K-0685, NASA contract number NAG-9-319, and the Progetto Finalizzato Robotica, CNR, contract 89.00493.PF.67.3. Claudio Melchiorri is a Research Associate on leave from DEIS, Dipartimento di Elettronica, Informatica e Sistemistica, Università di Bologna, Bologna, Italy. Support for Claudio Melchiorri as a Visiting Scientist at the A.I. Lab has been provided by a NATO-CNR joint fellowship program, grant No. 215.23/15.

1 Introduction

In the last few years, a growing interest has arisen in the field of manipulation and articulated hands. Major topics in this area have been the design and development of suitable mechanical devices, see [1, 2, 3, 4] for a few examples, and the development of techniques for the effective use of an articulated hand for grasping and manipulating objects, see [5, 6, 7] and many others.

Recently the attention of some researchers in this field has been focused on the redundant hand-arm system, and how to deal with the whole device. The approaches taken so far have dealt with the satisfaction of some optimality criteria, which consider a stable and feasible grasp as a major goal. [8, 9] In [8] the problem of the control of the whole device is separated into the two sub-problems of first choosing a suitable grasp pose for the hand, and then defining the arm position and orientation on this basis. In [9] the use of a non-linear programming technique, including several optimality criteria for the joints and the grasp, is proposed for the effective planning of the hand-arm trajectory.

Since the hand-arm system can be considered a redundant manipulator, it seems natural, in order to exploit fully the nature of the device, to investigate the results of research in the area of control of redundant manipulators in order to seek techniques that can be profitably used in this context.

The first observation is that the results presented in the field of redundant robots mainly deal with serial-link open-chain manipulators, i.e., devices constituted as a serial chain of links/joints. Redundancy is prevalently used for satisfying one or more secondary criteria, such as obstacle avoidance, singularity avoidance, optimization of task space indices or joint space criteria, while achieving the main task, for example the tracking of a planned trajectory for the end-effector.

On the other hand, there are some particularities, characterizing a hand-arm robotic system, which are relevant for the development of a suitable control strategy.

The first concerns the type of redundancy involved in the device. In [10] an interesting solution to the redundancy problem is presented: the proposed approach consists of considering a redundant arm as a multi-arm system in which non-redundant arms are serially connected together. The task of the end-effector is consequently separated into sub-tasks assigned to each of the sub-parts on the basis of the individual capabilities. This method, although original and interesting, cannot be integrally adopted in the present context since the device under consideration is not a serial robot. As a matter of fact, in our case a manipulator has a redundant parallel device -the hand- installed as an end-effector. Therefore, it is not possible to consider it as a serial mechanical chain.

A second comment may be made with respect to the "behaviors" which are expected from this type of mechanism. In fact, usually, both small and large motions are involved, requiring

A-1

different parts of the system to act in different ways. For example, in some circumstances it may not be desirable to move the arm, while in others the task may not be accomplished with only the limited motion capabilities of the fingers. On the other hand, the system cannot be generically considered as a macro/micro manipulator system [11, 12], because this "behavior" difference may not be needed, nor required, in other manipulation tasks.

A third characteristic of the system being considered is that, intrinsically, a hand-arm device has to interact physically with the environment. Therefore, any adopted control scheme must deal with the problem of force control along with the redundancy resolution.

In this work a method for exploiting the redundancy of a hand-arm mechanical system which addresses the above mentioned considerations is presented. The proposed technique uses the intrinsic capabilities of the arm and hand subsystems in terms of amplitude of motions, velocities, and degrees of precision for the achievement of a particular task. The technique, which relies on a kinematic inversion algorithm for redundant manipulators well-known in the literature, is based on a task-space description of the task, in terms of motions and/or forces applied to the object/environment. The algorithm, which is presented in literature as a closed-loop control scheme, uses as a central element the transpose $\mathbf{J}^T$ of the Jacobian matrix $\mathbf{J}$ of the manipulator. The basic form of the algorithm is modified here, adding constraints on the motions of the joints in the null space of $\mathbf{J}$ in order to take advantage of the differencies in the capabilities of the arm and the hand.

The paper is organized as follows. Section 2 gives a general background of the most popular techniques proposed in the literature for controlling a redundant manipulator. In Section 3 the basic scheme of the adopted kinematic inversion method is illustrated, while in section 4 the modifications are presented and discussed. Section 5 reports some simulations with a planar 4 degree-of-freedom redundant manipulator, while section 6 illustrates the results obtained with the implementation of the algorithm on our system, a Puma 560 carrying a Salisbury Hand. Section 7 concludes with some comments and plans for future activity.

2 Background

The chosen approach for controlling the hand-arm system has been to consider the hand-arm as a redundant manipulator, and to apply, with proper modifications, techniques adopted for redundant robots. As is well-known, one of the major problems in this field is represented by the solution of the kinematic equations, since it is not possible in general to solve them in a closed form.

In general, the kinematic problem can be stated as follows. Given the joint space Q, with dim(Q) = m, and the task space X, dim(X) = n, with n<m in the redundant case, the forward kinematics of a manipulator is expressed by a differentiable function:

$$\mathbf{f} : Q \rightarrow X,$$

which maps each set of joint angles $\mathbf{q}$ in Q into the corresponding position $\mathbf{x}$ in X:

$$\mathbf{x} = \mathbf{f}(\mathbf{q}). \tag{1}$$

Accordingly, an inverse kinematic function gives a joint-space vector $\mathbf{q}$ for each $\mathbf{x}$ in the work-space of the manipulator, i.e.

$$\mathbf{q} = \mathbf{f}^{-1}(\mathbf{x}), \tag{2}$$

such that $\mathbf{f}(\mathbf{q}) = \mathbf{x}$.

Often, only translational displacements are considered in X, and therefore n$\leq$3, with $X = R^n$, while in the general case both translational and rotational displacements are taken into account, and therefore n$\leq$6, with $X = R^{n_l} \cup R^{n_r}$, n = $n_l + n_r$, $n_l \leq 3$, $n_r \leq 3$.

Certainly, the most immediate approach to the solution of the inverse kinematics problem is to determine a closed-form solution of (2), [13], but, unfortunately, it is not possible to compute such a solution for every manipulator [14]. The inversion algorithms which are proposed in literature mostly fall into one of two major categories: (a) global (or path) inversion methods, and (b) local inversion methods. A global performance criterion is minimized in the first class of techniques, while locally optimal solutions are sought in the second one. However, the local optimization approach is the most widely adopted, since the global inversion methods are mainly limited to off-line trajectory planning [15, 16].

The local inversion methods are based on the differential relationship

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{3}$$

derived from (1), where $\dot{\mathbf{q}}$ are the joint velocities, $\dot{\mathbf{x}}$ the task space velocities, and $\mathbf{J} = \delta\mathbf{f}/\delta\mathbf{q}$ is the Jacobian matrix of $\mathbf{f}(\mathbf{q})$. Eq. (3) is then usually solved using a generalized inverse, or the Moore Penrose inverse [17], $\mathbf{J}^+$ of the Jacobian matrix $\mathbf{J}$.

One of the proposed method for solving (3) consists of expressing the solution $\dot{\mathbf{q}}$ as

$$\dot{\mathbf{q}} = \mathbf{J}(\mathbf{q})^+\dot{\mathbf{x}} + \alpha[\mathbf{I} - \mathbf{J}(\mathbf{q})^+\mathbf{J}(\mathbf{q})]\nabla H(\mathbf{q}) \tag{4}$$

where the first term on the right hand side of (4) is the minimum-norm least-squares solution of (3) (once an appropriate metric is defined in the Q and X spaces [18, 19]), while the second term represents a vector in $Null(\mathbf{J})$, the null space of $\mathbf{J}$, which is used for satisfying one or more secondary criteria, i.e. to optimize the differentiable objective function $H(\mathbf{q})$. Examples of such secondary criteria include: avoidance of obstacles, joint limits or singularities; optimization of task space indexes (such as manipulability ellipsoids, dexterity measures, ...), or joint space criteria (torque/velocity), and several others. See [20] for a general overview of the most commonly adopted criteria.

A conceptually different way for redundancy resolution has been proposed in [21] and more recently reformulated in [22]. In this approach, redundancy is used to accomplish an additional constraint task along with the original one. With this method, known as task-space augmentation or extended Jacobian technique, one must specify an additional task, expressed as a proper function of the joint variables, $\mathbf{h}(\mathbf{q}) = 0$. The solution is then computed in terms of the extended Jacobian $\mathbf{J} = [(\delta\mathbf{f}/\delta\mathbf{q})^T(\delta\mathbf{h}/\delta\mathbf{q})^T]^T$.

Another relevant approach has been proposed in [23, 24]: the task-priority-strategy. In this approach, a low-priority task is fulfilled in the null space of a higher priority task, solving in this way possible conflicts between different tasks.

The previously reported methods are based on a generalized inverse of the Jacobian matrix $\mathbf{J}$ (or on its pseudoinverse), and give the solution in terms of the joint velocities $\dot{\mathbf{q}}$. Therefore, in order to effectively solve the inverse kinematic equation (2), one must integrate $\dot{\mathbf{q}}$ to obtain the joint positions $\mathbf{q}$. This is usually done directly, in an open-loop fashion, possibly leading to non-accurate solutions $\mathbf{q}$ because of the linearization performed with the introduction of the Jacobian matrix. Moreover, since the core of this method is a differential relationship, the resulting solution fails if appropriate initial conditions are not provided.

A solution to these inconveniences has been found in reformulating the problem in terms of a closed-loop scheme [25, 26]. In [25] two closed-loop schemes are proposed for the solution of (3), one based on $\mathbf{J}^+$ and the other on the transpose of the Jacobian matrix, $\mathbf{J}^T$. In [26], the latter scheme is independently proposed, as a general technique for solving, with some very general limitations, any set of nonlinear equations. This technique has been recently improved and extended to different cases, see [27]-[34]. Here, the main idea is to reformulate the inverse kinematic problem in terms of the convergence and stability of an equivalent closed-loop control system. This leads to the possibility of effectively solving the inverse kinematic problem for redundant manipulator in a robust and accurate way.

The scheme, referred to here as the *Jacobian transpose method*, [34], Fig. 1, also has other interesting properties. A first feature is that it requires only the computation of the forward kinematic functions $(\mathbf{f}(\mathbf{q}), \mathbf{J}(\mathbf{q}))$, avoiding, in its basic formulation, the generalized

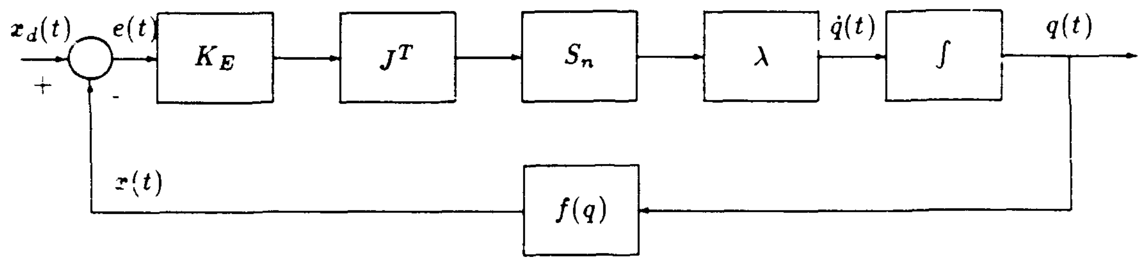


Figure 1: The basic scheme of the Jacobian transpose method.

inverse of the Jacobian matrix. This leads to a reliable scheme without numerical problems and instabilities such as those related to singular configurations of the manipulator. Moreover, the stability of the scheme may be easily demonstrated using a Lyapunov analysis. In the continuous time domain, it can be proven that the tracking error may be arbitrarily reduced with an high gain λ. In discrete time, a compromise has to be accepted between the convergence velocity and the stability of the algorithm [29, 34]. An additional interesting feature is that it is very simple to add constraints on the joint motions for the achievement of desired "behaviors" of the system. Finally, besides generating joint positions, the scheme provides also joint velocities $\dot{\mathbf{q}}(t)$, and, with minor modifications, also joint accelerations $\ddot{\mathbf{q}}(t)$, [26].

Because of the above reported considerations, it was decided, in this first stage of work with the hand-arm, to adopt an inversion kinematic scheme based on this technique for the kinematic control of the system.

3 The basic algorithm

The basic scheme of the inverse Jacobian algorithm, as proposed in [29, 33], is shown in Fig. 1. In the Figure, $\mathbf{x}_d(t)$ is a desired trajectory, $\mathbf{x}(t)$ is the actual trajectory of the manipulator, $\dot{\mathbf{q}}(t)$ and $\mathbf{q}(t)$ are the joint velocities and positions respectively, $\mathbf{J}$ is the Jacobian, $\mathbf{K}_E$ is a stiffness matrix, and $\lambda(> 0)$ is a gain which affects the convergence velocity of the algorithm itself.

An interesting interpretation of the scheme, which helps to give a physical insight into the technique, is the following. It is well-known that the static relationship between the forces $\mathbf{F}$ applied at the end-effector and the joint torques τ is given by

$$\tau = \mathbf{J}^T\mathbf{F}.$$

The computation of the joint velocities $\dot{\mathbf{q}}$ in the scheme, see Fig. 1, may therefore be related to the generation of a restoring force $\mathbf{F} = \lambda \mathbf{K}_E \mathbf{e}(t)$ due to a positional error of an ideal manipulator, with null mass and unit viscous damping factor.

In the continuous time domain, the proof of the convergence of the joint positions $\mathbf{q}(t)$ to a set $\mathbf{q}_d(t)$ such that $\mathbf{f}(\mathbf{q}_d) = \mathbf{x}_d$ is quite straightforward. Given the tracking error defined as

$$\mathbf{e}(t) = \mathbf{x}_d(t) - \mathbf{x}(t), \tag{5}$$

and defining a Lyapunov function as

$$V(\mathbf{e}) = \frac{\mathbf{e}^T K_E^T \mathbf{e}}{2} > 0, \tag{6}$$

then

$$\dot{V}(\mathbf{e}) = \mathbf{e}^T K_E^T \dot{\mathbf{e}} = \mathbf{e}^T \mathbf{K}_E^T (\dot{\mathbf{x}}_d - \mathbf{J}\dot{\mathbf{q}}) \tag{7}$$

in which the dependency of the functions from the time t and the joint positions $\mathbf{q}$ is omitted. With the choice

$$\dot{\mathbf{q}} = [\lambda + \frac{(\mathbf{e}^T \mathbf{K}_E \dot{\mathbf{x}}_d)}{(\mathbf{e}^T \mathbf{K}_E^T \mathbf{J}\mathbf{J}^T \mathbf{K}_E \mathbf{e})}] \mathbf{J}^T \mathbf{K}_E \mathbf{e}, \ \lambda > 0, \tag{8}$$

it is easy to see how (7) may be made negative definite. This implies that $\mathbf{e}(t) \rightarrow 0$ and therefore $\mathbf{q}(t) \rightarrow \mathbf{q}_d(t)$. In [29] it is pointed out how eq. (8) may be, for computational convenience, simplified to

$$\dot{\mathbf{q}} = \lambda \mathbf{J}^T \mathbf{K}_E \mathbf{e}, \tag{9}$$

allowing the function $\dot{V}$ to be negative-definite only outside a region of the error space containing the stability point $\mathbf{e} = 0$. With the choice (9), the maximum tracking error is directly related to $\dot{\mathbf{x}}_d$ and inversely to λ, while in steady state, since $\dot{\mathbf{x}}_d = 0$, the tracking error will be zero. In this situation, an increase in the gain λ results into a reduction of the tracking error, which may therefore be arbitrarily reduced [28, 31].

A discrete time stability proof of the algorithm is presented in [34]. In this work it is also observed how this technique can be related to the context of non linear optimization.

As a matter of fact, the algorithm may be interpreted as the steepest descent method, a well-known technique in the area of multidimensional optimization problems [35]. One of the major modifications in the discrete time version of the algorithm is that, in order both to obtain the maximum convergence rate for the scheme and to avoid instability problems, the gain λ has to be updated at each sampling period T. In fact, given the discrete time version of the Lyapunov function (6) at $t = nT$

$$V_n = \frac{\mathbf{e}_n^T K_E^T \mathbf{e}_n}{2}$$

and computing the joint velocities as

$$\dot{\mathbf{q}}_n = \lambda_n \mathbf{J}_n^T \mathbf{K}_E \mathbf{e}_n, \tag{10}$$

the difference $V_{n+1} - V_n$ may be made negative with the choice

$$\lambda_n = \frac{1}{T} \frac{\mathbf{e}_n^T \mathbf{K}_E^T \mathbf{J}_n \mathbf{S}_n \mathbf{J}_n^T \mathbf{K}_E \mathbf{e}_n}{\mathbf{e}_n^T \mathbf{K}_E^T \mathbf{J}_n \mathbf{S}_n \mathbf{J}_n^T \mathbf{K}_E \mathbf{J}_n \mathbf{S}_n \mathbf{J}_n^T \mathbf{K}_E \mathbf{e}_n} \tag{11}$$

where $\mathbf{S}_n$ is a diagonal matrix whose elements are properly computed to limit the maximum values of τ, (see Fig. 1), [34].

A final remark on the characteristics of this scheme concerns the possibility of getting "stuck", i.e. to generate a null joint velocity vector, $\dot{\mathbf{q}} = 0$, also if $\mathbf{e} \neq 0$. This happens when $\mathbf{K}_E\mathbf{e} \in Null(\mathbf{J}^T)$. However, this does not represent a serious limitation to the applicability of the algorithm. As a matter of fact, besides being an easily detectable condition ($\dot{\mathbf{q}} = 0$; $\mathbf{e} \neq 0$), it can be argued that the term $\mathbf{K}_E\mathbf{e}$ can be easily modified in such a way that $\mathbf{K}_E\mathbf{e} \notin Null(\mathbf{J}^T)$. This is performed by adding suitable constraints on the joint space (as done in [34]), or in the task space. Obviously, in this latter case if the trajectory has some components which are constantly in $Null(\mathbf{J}^T)$, the algorithm will not be able to compensate for errors in them.

4 The adopted algorithm

In order to apply the inversion technique outlined in the previous section, some modifications are necessary to address the particularities of the hand-arm system we are considering. The first problem is that the method, in its basic form, does not make any distinction among the various joints of the arm and of the hand. This is not acceptable, since, as mentioned in the Introduction, there are tasks in which a different action is desired from the two subsystems.

For example, in order to quickly approach an object, one could take advantage of the fast movements of one part of the system, say the hand, while a slower motion is executed by the arm, restoring at the end of its motion the hand to an appropriate position for an optimal grasp of the object. In other circumstances, the motion of the hand is not needed, nor desired: if an object is stably grasped, it could be desirable for the arm alone to generate the motion of the hand/object in the environment. Therefore, in different circumstances, a "difference" in the "behaviors" of the two sub-systems is required. As a consequence, the algorithm has to be modified in order to be able to adapt the joint position/velocity values of the two sub-systems to the different requirements of the task being performed. Another capability, which may be of interest, is the possibility of maintaining some joints (for example the joints of the hand) both far from critical positions (singularities, joint limits) and close to desired ones (suitable for optimal grasp).

In the following, let us indicate with the subscript "S" the quantities of the whole system, for example the Jacobian $\mathbf{J}_S$, the joint velocities $\dot{\mathbf{q}}_S$, etc., while "P", "H" and "F" denote the arm (Puma), the hand, and a finger respectively. For the system, eq. (3) becomes now

$$^B\dot{\mathbf{x}}_S = {}^B\mathbf{J}_S\dot{\mathbf{q}}_S = \left[\begin{array}{cc} ^B\mathbf{J}_P & ^B\mathbf{J}_H \end{array}\right]\left[\begin{array}{c} \dot{\mathbf{q}}_{PS} \\ \dot{\mathbf{q}}_{HS} \end{array}\right], \tag{12}$$

where the superscript "B" indicates that the quantities are expressed in the base frame $\mathcal{B}$, Fig. 2.

The modifications which it may be necessary to introduce into the solution given by the basic algorithm, shown in Fig. 1, must not interfere with the main task of the manipulator. This is accomplished if the modifications operate in the null space of the Jacobian $\mathbf{J}_S$, with the additional advantage that the stability proofs of the algorithm, given in the previous section, still apply. Hence, the final set of joint velocities $\dot{\mathbf{q}}_S$ may be thought to be composed of two terms:

$$\dot{\mathbf{q}}_S = \dot{\mathbf{q}}_A + \dot{\mathbf{q}}_N = \lambda\, {}^B\mathbf{J}_S^T\mathbf{K}_E\mathbf{e} + \dot{\mathbf{q}}_N$$

where $\dot{\mathbf{q}}_N \in Null(\mathbf{J}_S)$, and $\dot{\mathbf{q}}_A$ is the solution computed by the basic Jacobian transpose algorithm. Obviously, since $\dot{\mathbf{q}}_N = \left[\begin{array}{cc} \dot{\mathbf{q}}_{PN}^T & \dot{\mathbf{q}}_{HN}^T \end{array}\right]^T \in Null(\mathbf{J}_S)$, the following relationship must hold:

$$0 = {}^B\mathbf{J}_P\dot{\mathbf{q}}_{PN} + {}^B\mathbf{J}_H\dot{\mathbf{q}}_{HN}.$$

In the following subsections, the determination of the term $\dot{\mathbf{q}}_N$ is discussed, considering for simplicity only one finger of the hand and only linear displacements for the end effector.

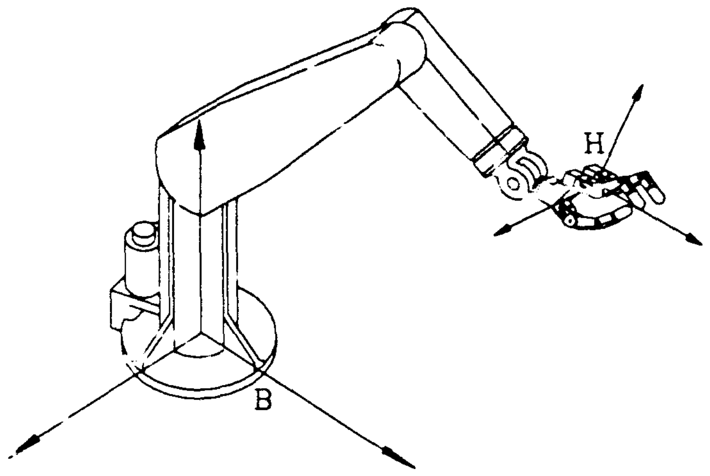


Figure 2: The Puma - Salisbury Hand system.

Three basic cases are considered, namely the execution of a task involving: (a) only the motion of the finger joints; (b) only the motion of the arm; (c) the motion of the whole system including, during the last part of the trajectory, the achievement of the desired position of the finger.

4.1 Generating motion only with the finger.

In this case it is required that the joints of the arm be maintained at a constant value ($\mathbf{q}_{Pinit}$), i.e. that $\dot{\mathbf{q}}_{PS} = \mathbf{0}$, and that the finger joints set-points be computed in such a way to follow a desired task space trajectory $\mathbf{x}_d$. The final solution of (12) is then in the form

$$\dot{\mathbf{q}}_S = \begin{bmatrix} \dot{\mathbf{q}}_{PS} \\ \dot{\mathbf{q}}_{FS} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{q}_{FS} \end{bmatrix} \tag{13}$$

with

$$\mathbf{x}_d(t) = \mathbf{f}(\begin{bmatrix} \mathbf{q}_{Pinit}^T & \mathbf{q}_{FS}^T(t) \end{bmatrix}^T). \tag{14}$$

The joint velocities $\dot{\mathbf{q}}_N$ may then be computed from

$$\begin{aligned} \dot{\mathbf{q}}_{PS} &= \dot{\mathbf{q}}_{PA} + \dot{\mathbf{q}}_{PN} = 0 \\ 0 &= {}^B\mathbf{J}_P \dot{\mathbf{q}}_{PN} + {}^B\mathbf{J}_F \dot{\mathbf{q}}_{FN} \end{aligned}$$

which gives

$$\begin{aligned}\dot{\mathbf{q}}_{PN} &= -\dot{\mathbf{q}}_{PA}\\ \dot{\mathbf{q}}_{FN} &= {}^B\mathbf{J}_F^+\,{}^B\mathbf{J}_P\dot{\mathbf{q}}_{PA}\end{aligned}$$

or, in matrix form,

$$\dot{\mathbf{q}}_{N1} = \begin{bmatrix} -\dot{\mathbf{q}}_{PA} \\ {}^B\mathbf{J}_F^+\,{}^B\mathbf{J}_P\dot{\mathbf{q}}_{PA} \end{bmatrix} = \begin{bmatrix} -\mathbf{I}_p & 0 \\ {}^B\mathbf{J}_F^+\,{}^B\mathbf{J}_P & 0 \end{bmatrix}\dot{\mathbf{q}}_A = \mathbf{P}_F\dot{\mathbf{q}}_A. \tag{15}$$

where $\mathbf{I}_p$ is the $n_p x n_p$ identity matrix. The matrix $\mathbf{P}_F$ in (15) generates, given a solution $\dot{\mathbf{q}}_A$, a joint velocity term $\dot{\mathbf{q}}_N$ in $Null(\mathbf{J}_S)$ which, added to $\dot{\mathbf{q}}_A$, verifies the two conditions expressed by (13) and (14). The matrix $\mathbf{P}_F$ may therefore be considered as a projector, clearly not orthogonal, of the given solution $\dot{\mathbf{q}}_A$ in the null space of $\mathbf{J}_S$.

4.2 Generating motion only with the arm.

It is now required that the joints of the arm be in charge of moving the manipulator along a specified trajectory, while $\dot{\mathbf{q}}_{FS} = \mathbf{0}$, i.e. $\mathbf{q}_F(t) = \mathbf{q}_{Finit}$. The solution of (12) is, in this case, in the form

$$\dot{\mathbf{q}}_S = \begin{bmatrix} \dot{\mathbf{q}}_{PS} \\ \dot{\mathbf{q}}_{FS} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}}_{PS} \\ 0 \end{bmatrix} \tag{16}$$

with

$$\mathbf{x}_d(t) = \mathbf{f}(\begin{bmatrix} \mathbf{q}_{PS}^T(t) & \mathbf{q}_{Finit}^T \end{bmatrix}^T). \tag{17}$$

The joint velocities $\dot{\mathbf{q}}_N$ are now computed from

$$\begin{aligned} 0 &= {}^B\mathbf{J}_P\dot{\mathbf{q}}_{PN} + {}^B\mathbf{J}_F\dot{\mathbf{q}}_{FN}\\ \dot{\mathbf{q}}_{FS} &= \dot{\mathbf{q}}_{FA} + \dot{\mathbf{q}}_{FN} = 0\end{aligned}$$

which gives

$$\begin{aligned}\dot{\mathbf{q}}_{PN} &= {}^B\mathbf{J}_P^+\,{}^B\mathbf{J}_F\dot{\mathbf{q}}_{FA}\\ \dot{\mathbf{q}}_{FN} &= -\dot{\mathbf{q}}_{FA}\end{aligned}$$

and, in matrix form,

$$\dot{\mathbf{q}}_{N2} = \begin{bmatrix} {}^B\mathbf{J}_P^+ \, {}^B\mathbf{J}_F \dot{\mathbf{q}}_{PA} \\ -\dot{\mathbf{q}}_{FA} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & {}^B\mathbf{J}_P^+ \, {}^B\mathbf{J}_F \\ \mathbf{0} & -\mathbf{I}_f \end{bmatrix} \dot{\mathbf{q}}_A = \mathbf{P}_P \dot{\mathbf{q}}_A. \tag{18}$$

where $\mathbf{I}_p$ is the $n_f x n_f$ identity matrix. The matrix $\mathbf{P}_P$ in (18) generates, given a solution $\dot{\mathbf{q}}_A$, a joint velocity term $\dot{\mathbf{q}}_N$ in $Null(\mathbf{J})$ which, added to $\dot{\mathbf{q}}_A$, verifies the two conditions expressed by (16) and (17). The matrix $\mathbf{P}_P$, similarly to $\mathbf{P}_F$, may be considered a projector of the given solution $\dot{\mathbf{q}}_A$ in the null space of $\mathbf{J}_S$.

4.3 Moving the finger to a desired position.

The actions to be determined now are intended to achieve the joint position vector $\mathbf{q}_F$ to a desired value $\mathbf{q}_{Fd}$, while the whole system is following a specified trajectory $\mathbf{x}_d$. A practical way for the achievement of this goal is to generate a joint velocity term which compensates for the positional errors $(\mathbf{q}_{Fd} - \mathbf{q}_F)$. In other words, the term $\dot{\mathbf{q}}_N$ may now be computed from

$$\begin{aligned} 0 &= {}^B\mathbf{J}_P \dot{\mathbf{q}}_{PN} + {}^B\mathbf{J}_F \dot{\mathbf{q}}_{FN} \\ \dot{\mathbf{q}}_{FN} &= \mathbf{K}(\mathbf{q}_{Fd} - \mathbf{q}_F) \end{aligned}$$

from which

$$\begin{aligned} \dot{\mathbf{q}}_{PN} &= -{}^B\mathbf{J}_P^+ \, {}^B\mathbf{J}_F \mathbf{K}(\mathbf{q}_{Fd} - \mathbf{q}_F) \\ \dot{\mathbf{q}}_{FN} &= \mathbf{K}(\mathbf{q}_{Fd} - \mathbf{q}_F) \end{aligned}$$

or, in matrix form,

$$\dot{\mathbf{q}}_{N3} = -\begin{bmatrix} \mathbf{0} & {}^B\mathbf{J}_P^+ \, {}^B\mathbf{J}_F \\ \mathbf{0} & -\mathbf{I}_f \end{bmatrix} \mathbf{K}(\mathbf{q}_{Fd} - \mathbf{q}_F) = -\mathbf{P}_P \mathbf{K}(\mathbf{q}_{Fd} - \mathbf{q}_F) \tag{19}$$

in which the projector $\mathbf{P}_P$ is the same as in (18).

The three terms $\dot{\mathbf{q}}_{Ni}$ in eq. (15), (18), (19), may be combined together, resulting in the scheme shown in Fig. 3. In the scheme, the three factors α_1, α_2, and α_3 ($0 \leq \alpha_i \leq 1$, i=1,2,3), are variable gains which are used to modulate the actions of the three terms $\dot{\mathbf{q}}_{N1}$, $\dot{\mathbf{q}}_{N2}$, and $\dot{\mathbf{q}}_{N3}$ on the original solution $\dot{\mathbf{q}}_A$. By properly changing in real time the three factors

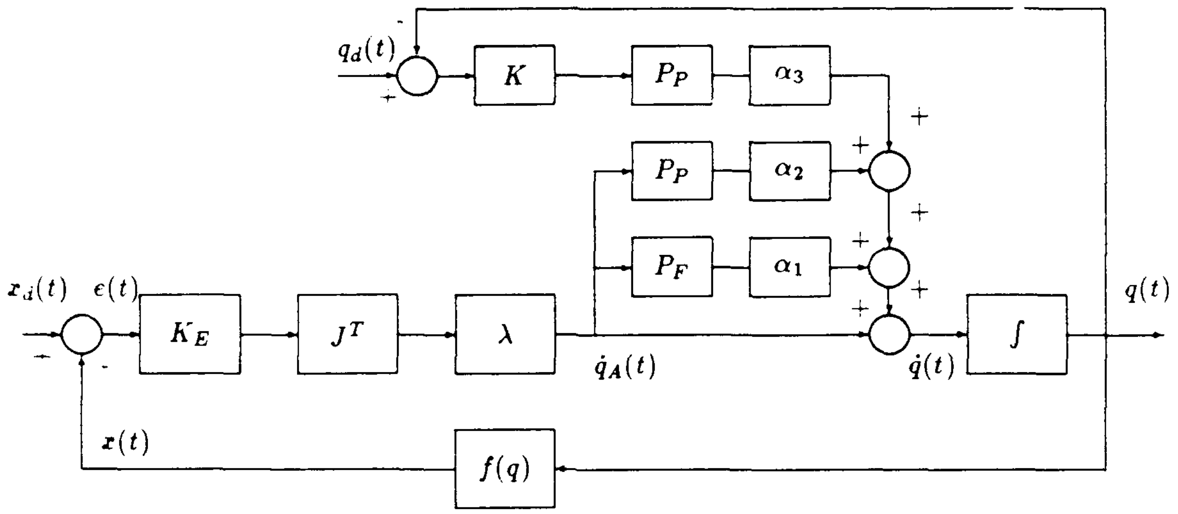


Figure 3: The modified Jacobian transpose scheme.

α_i, it is possible to modulate the possible "behaviors" of the system. If $\alpha_i = 0$, the relative constraint term $\dot{\mathbf{q}}_{Ni}$ is neglected, while when $\alpha_i = 1$, the constraint is fully active.

A final remark may be made with respect to the projectors $\mathbf{P}_F$ and $\mathbf{P}_P$. As a matter of fact in these matrices a term of the type $\mathbf{J}_A^+\mathbf{J}_B$ is present (the sub-matrix ${}^B\mathbf{J}_F^+\,{}^B\mathbf{J}_P$ in $\mathbf{P}_F$ or the sub-matrix ${}^B\mathbf{J}_P^+\,{}^B\mathbf{J}_F$ in $\mathbf{P}_P$). In the two cases, this implies that the original trajectory is tracked without errors iff $\mathbf{J}_B\dot{\mathbf{q}}_B \in Range(\mathbf{J}_A)$. If $Range(\mathbf{J}_A) = Range(\mathbf{J}_B)$, the introduction of the projectors does not change the trajectory of the system, otherwise, only the component in $Range(\mathbf{J}_A)$ may be compensated for with this solution. If only one finger is considered, this problem may become relevant only with the use of the matrix $\mathbf{P}_F$, since (with the exceptions of the singularities) the range space of the arm is the whole $\mathcal{R}^6$.

4.4 Computation of the gains α_i.

An interesting problem is how to define suitable strategies for the computation in real-time of the scalars α_i. The solution $\dot{\mathbf{q}}_S$ is now expressed as

$$\dot{\mathbf{q}}_S = \dot{\mathbf{q}}_A + \alpha_1\dot{\mathbf{q}}_{N1} + \alpha_2\dot{\mathbf{q}}_{N2} + \alpha_3\dot{\mathbf{q}}_{N3}.$$

It is not meaningful to have $\alpha_1 = \alpha_2 = \alpha_3 = 1$, since this would imply the simultaneous activation of opposing constrains on the solution, while the choice $\alpha_1 = \alpha_2 = \alpha_3 = 0$ means that the original solution $\dot{\mathbf{q}}_A$ is adopted. In particular, the choice $\alpha_1 = \alpha_2 = 1$ implies that the two contradictory constraints of keeping both the arm and the hand blocked while following a trajectory are imposed. As a matter of fact, this choice results in a 'switching' of the motions produced in the end-effector by the the two subsystems, and the final effect is simply to have a different set of joint velocities $\dot{\mathbf{q}}_S$.

As previously mentioned, it may be convenient to take advantage, at least during the first period of motion, of the fastest part of the system. It seems therefore reasonable to have, at the beginning of a task, the gains set as

$$\alpha_1 = 1, \; \alpha_2 = \alpha_3 = 0.$$

These values have to be changed when, for example, the finger's joints are close to a limit, or when the tracking error $||\mathbf{e}||$ is greater than a maximum allowed value $||\mathbf{e}||_{max}$. A way to take these constraints into consideration is to compute the gains as

$$\begin{aligned} \alpha_1 &= \begin{cases} 1 & \text{if } -q_{Ti} < q_i < q_{Ti}, i = 1, ..., n_f; \text{ and } ||\mathbf{e}|| < ||\mathbf{e}||_{max} \\ e^{-\mu\Delta q - \nu\Delta e} & \text{otherwise;} \end{cases} && (20) \\ \alpha_2 &= (1-\alpha_1)\beta && (21) \\ \alpha_3 &= (1-\alpha_1)\gamma && (22) \end{aligned}$$

where n_f is the number of joints of the finger, q_{Ti} a threshold value which limits the motion of the i-th joint , and q_{Li} the actual joint limit, $\Delta q = max\frac{||q_i - q_{Ti}||}{||q_i - q_{Li}||}$, $\Delta e = ||\mathbf{e}|| - ||\mathbf{e}_{max}||$, μ, ν two positive scalars which are set to 1 if the relative limit is broken. In this way the finger takes care of the required motion, provided that all the joints, as well as the tracking error, are within proper thresholds. If one of these two conditions fails, with the choices (20)-(22) the arm starts moving, while the finger achieves the desired configuration.

If other "behaviors" of the system are desired, different combinations of the three gains are possible. For example, if an object has been grasped, and there is no need of modifying the grasp pose, α_2 is set to 1, and all the movements of the system are generated by the arm.

4.5 The force feedback loop.

The final aspect of the problem we deal with is the need to consider, in the generation of the joint trajectories, the forces that are applied to the environment. To this purpose, a further feedback loop is added to the scheme, leading to the diagram shown in Fig. 4. In this scheme, $\mathbf{F}_d$ is a desired force, $\mathbf{e}_F = (\mathbf{F}_d - \mathbf{F})$ is the force error, and $\mathbf{K}_F$ a compliance matrix which represents a model of the manipulator-environment interaction. The stability of this loop is now affected by the manipulator dynamics and its control system. Considering an ideal system, i.e. assuming the forward kinematics $\mathbf{f}(\mathbf{q})$ as model of the manipulator/control

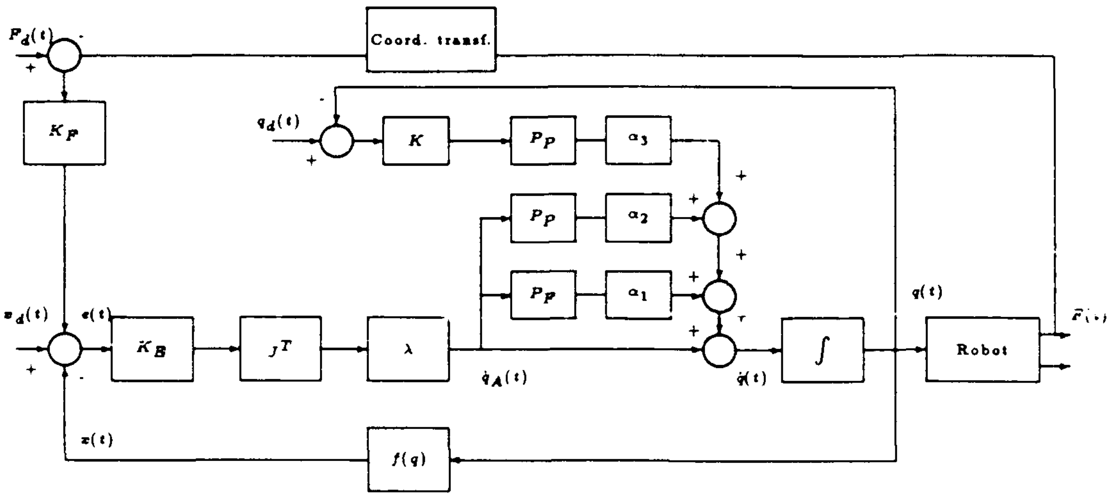


Figure 4: The force feedback loop in the Jacobian transpose scheme.

system, and a stiffness matrix $\mathbf{K}_F$ for modeling the interaction with the environment, the stability proof follows the same line as before. This model will be used in the following discussions.

4.6 Extension to the whole hand.

When the whole hand is taken into consideration, it is necessary to consider an extended task space. In fact, we are interested now not only in the specification of the object position/orientation, but also, once the object is grasped, in the relative displacements of the fingers, i.e. in the distances between the fingertips. Considering three fingers, the new forward kinematic function is therefore

$$\mathbf{x}(\mathbf{q}) = \begin{bmatrix} \mathbf{f}(\mathbf{q}) \\ \mathbf{d}(\mathbf{q}) \end{bmatrix} \in \mathcal{R}^9 \tag{23}$$

where $\mathbf{f}(\mathbf{q}) \in \mathcal{R}^6$ are the forward kinematic equations relating the hand-arm joints values to the position/orientation of the object, and $\mathbf{d}(\mathbf{q}) \in \mathcal{R}^3$ are the relative displacements of the fingertips. It is easy to see how only the joints of the hand affect this latter vector. Assuming a grasp on a rigid object, and without slip at the contact points, the computation of these two functions is quite straightforward. In particular, the function $\mathbf{d}(\mathbf{q})$ results as the magnitude of the difference of the positional vectors of the three fingertips, and, how already pointed out, is a function only of the hand joints. We are now interested in defining a differential relationship between the set of joint velocities and the set of object velocities and "internal velocities", i.e. the relative velocities of the fingertips.

In [1] the notion of grasp matrix $\mathbf{G}$ has been introduced. This matrix is a 9x9 matrix which relates the forces and displacements of the object to the forces and displacements of the single fingers, i.e.

$$\mathbf{F} = \mathbf{G}^T \mathcal{F}$$

or, in the velocity domain,

$$\dot{\mathbf{x}}_f = \mathbf{G}^{-1} \dot{\mathbf{x}}_o \tag{24}$$

where:
$\mathbf{F} = \left[\begin{array}{ccc} \mathbf{F}_1^T & \mathbf{F}_2^T & \mathbf{F}_3^T \end{array} \right]^T$ represents the forces applied at the fingertips
$\mathcal{F} = \left[\begin{array}{ccccc} \mathbf{f}^T & \mathbf{t}^T & f_{12} & f_{23} & f_{31} \end{array} \right]^T$ are the 6 external and 3 internal forces acting on the object
$\dot{\mathbf{x}}_f = \left[\begin{array}{ccc} \dot{\mathbf{x}}_{f1}^T & \dot{\mathbf{x}}_{f2}^T & \dot{\mathbf{x}}_{f3}^T \end{array} \right]^T$ are the three linear velocities of the fingertips
$\dot{\mathbf{x}}_o = \left[\begin{array}{ccccc} \mathbf{v}_o^T & \omega_o^T & \dot{d}_{12} & \dot{d}_{23} & \dot{d}_{31} \end{array} \right]^T$ are the 6 velocities of the object and the relative velocities of the three contact points, expressed as function of the hand joints only.

Eq. (24) may be written as

$${}^H\dot{\mathbf{x}}_f = \begin{bmatrix} {}^H\mathbf{J}_1\dot{\mathbf{q}}_1 \\ {}^H\mathbf{J}_2\dot{\mathbf{q}}_2 \\ {}^H\mathbf{J}_3\dot{\mathbf{q}}_3 \end{bmatrix} = \begin{bmatrix} {}^H\mathbf{J}_1 & 0 & 0 \\ 0 & {}^H\mathbf{J}_2 & 0 \\ 0 & 0 & {}^H\mathbf{J}_3 \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_1 \\ \dot{\mathbf{q}}_2 \\ \dot{\mathbf{q}}_3 \end{bmatrix} = {}^H\mathbf{J}'_H \dot{\mathbf{q}}_H = \mathbf{G}^{-1}\, {}^H\dot{\mathbf{x}}_o$$

then

$${}^H\dot{\mathbf{x}}_o = \mathbf{G}\, {}^H\mathbf{J}'_H \dot{\mathbf{q}}_H = {}^H\mathbf{J}_H \dot{\mathbf{q}}_H = \begin{bmatrix} {}^H\mathbf{J}_{Hf} \\ {}^H\mathbf{J}_{Hd} \end{bmatrix} \dot{\mathbf{q}}_H \tag{25}$$

in which all the quantities are expressed in the hand frame $\mathcal{H}$, see Fig. 2, and the two terms ${}^H\mathbf{J}_{Hf} = \left[\begin{array}{cc} {}^H\mathbf{J}_{Hv}^T & {}^H\mathbf{J}_{H\omega}^T \end{array} \right]^T$ and ${}^H\mathbf{J}_{Hd}$ refer respectively to the object velocity and to the "internal velocity", i.e. the deformations of the grasp triangle, the imaginary triangle between the three contact points. Equation (25) may be expressed in the base frame as

$${}^B\dot{\mathbf{x}}_o = \begin{bmatrix} {}^B\mathbf{R}_H & {}^B\mathbf{R}_H \mathbf{P}_{O\otimes} & 0 \\ 0 & {}^B\mathbf{R}_H & 0 \\ 0 & 0 & {}^B\mathbf{R}_H \end{bmatrix} {}^H\mathbf{J}_H \dot{\mathbf{q}}_H \tag{26}$$

in which ${}^B\mathbf{R}_H$ is the 3x3 rotational matrix expressing the rotation between the frames H and B, and $\mathbf{P}_{O\otimes}$ is a skew symmetric matrix equivalent to the cross product $(\mathrm{x}\,{}^B\mathbf{p}_{obj})$, where the vector ${}^B\mathbf{p}_{obj}$ gives the object position in the base frame.

As far as the arm is concerned, a relation similar to (26) may be defined. The derivatives of the functions $\mathbf{f}(\mathbf{q})$ and $\mathbf{d}(\mathbf{q})$ with respect to the arm joints yield the Jacobian for the arm:

$$^{B}\mathbf{J}_{P} = \begin{bmatrix} \delta\mathbf{f}/\delta\mathbf{q}_{P} \\ \delta\mathbf{d}/\delta\mathbf{q}_{P} \end{bmatrix} = \begin{bmatrix} ^{B}\mathbf{J}_{Pf} \\ \mathbf{0} \end{bmatrix}$$

where the matrix $^{B}\mathbf{J}_{Pf}$ computes as follows

$$\mathbf{J}_{Pf} = \begin{bmatrix} (^{B}\mathbf{J}_{P_v} - \mathbf{P}_{X}\,^{B}\mathbf{J}_{P_\omega}) \\ ^{B}\mathbf{J}_{P_\omega} \end{bmatrix}$$

where $^{B}\mathbf{J}_{Pf} = \begin{bmatrix} ^{B}\mathbf{J}^{T}_{P_v} & ^{B}\mathbf{J}^{T}_{P_\omega} \end{bmatrix}^{T}$ is the arm Jacobian, in which the terms generating the linear and rotational velocities are in evidence. The matrix $\mathbf{P}_X$ is a skew symmetric matrix equivalent to the cross product $(\mathrm{x}^{B}\mathbf{p}_o)$, defined on the basis of the elements of the vector $^{B}\mathbf{p}_o$, giving the object position with respect to the hand frame, expressed in the base frame. Therefore, the final Jacobian $\mathbf{J}_S$ of the hand-arm system, in our case a 9x15 matrix, is

$$^{B}\mathbf{J}_{S} = \begin{bmatrix} ^{B}\mathbf{J}_{P} & ^{B}\mathbf{J}_{H} \end{bmatrix} = \begin{bmatrix} ^{B}\mathbf{J}_{Pf} & ^{B}\mathbf{J}_{Hf} \\ \mathbf{0} & ^{B}\mathbf{J}_{Hd} \end{bmatrix}$$

and the differential relationship between the set of object velocities and internal velocities and the set of joint velocities is

$$\dot{\mathbf{x}}_o = \begin{bmatrix} \mathbf{J}_{Pf} & \mathbf{J}_{Hf} \\ \mathbf{0} & \mathbf{J}_{Hd} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_P \\ \dot{\mathbf{q}}_H \end{bmatrix} = \mathbf{J}_S \dot{\mathbf{q}}_S$$

where the superscript B is omitted for brevity.

The Jacobian transpose method may still be applied, with the additional need of specifying the internal displacements $\mathbf{d}$. This implies that the solution $\dot{\mathbf{q}}_A$ given by the algorithm is

$$\begin{aligned} \dot{\mathbf{q}}_P &= \lambda \mathbf{J}^{T}_{P}\mathbf{K}_{Ef}\mathbf{e}_f \\ \dot{\mathbf{q}}_H &= \lambda(\mathbf{J}^{T}_{Hf}\mathbf{K}_{Ef}\mathbf{e}_f + \mathbf{J}^{T}_{Hd}\mathbf{K}_{Ed}\mathbf{e}_d) \end{aligned}$$

from which it is clear how the joints of the Puma may compensate only for errors in the position/orientation of the objects, while the joints of the fingers also affects the internal velocities. In this case, without the introduction of the projectors, if $\mathbf{d}(t)$ is maintained at a constant value and a movement of the object is required, both $\dot{\mathbf{q}}_P$ and $\dot{\mathbf{q}}_H$ are changed in order to follow the trajectory, with the additional requirement for the hand to maintain a constant grasp triangle.

In the following, the three cases of movements of the hand, of the arm, or changes in the relative displacements of the fingertips are considered.

4.6.1 Case 1: motion of an object grasped in the hand with only the joints of the fingers.

If a modification of the grasp triangle is not required at the task level, i.e. **d**=cost., the specified trajectory originates a velocity vector of the form $\dot{\mathbf{x}}_o = \begin{bmatrix} \dot{\mathbf{x}}_f^T & \mathbf{0}^T \end{bmatrix}^T$, therefore

$$\begin{bmatrix} \dot{\mathbf{x}}_f \\ \mathbf{0} \end{bmatrix} = \mathbf{J}_S \dot{\mathbf{q}}_S.$$

If it is desired to move the grasped object by using only the hand, it follows that

$$\begin{aligned} \dot{\mathbf{q}}_{PS} = \dot{\mathbf{q}}_{PA} + \dot{\mathbf{q}}_{PN} &= \mathbf{0} \\ \mathbf{J}_S \dot{\mathbf{q}}_N &= \mathbf{0} \end{aligned}$$

therefore

$$\begin{aligned} -\mathbf{J}_{Pf}\dot{\mathbf{q}}_{PA} + \mathbf{J}_{Hf}\dot{\mathbf{q}}_{HN} &= 0 \\ \mathbf{J}_{Hd}\dot{\mathbf{q}}_{HN} &= 0 \end{aligned}$$

from which

$$\dot{\mathbf{q}}_{HN} = \mathbf{J}_{Hf}^{+}\mathbf{J}_{Pf}\dot{\mathbf{q}}_{PA}$$

or

$$\dot{\mathbf{q}}_N = \begin{bmatrix} -\dot{\mathbf{q}}_{PA} \\ \mathbf{J}_{Hf}^{+}\mathbf{J}_{Pf}\dot{\mathbf{q}}_{PA} \end{bmatrix} = \begin{bmatrix} -\mathbf{I}_p & \mathbf{0} \\ \mathbf{J}_{Hf}^{+}\mathbf{J}_{Pf} & \mathbf{0} \end{bmatrix} \dot{\mathbf{q}}_A = \mathbf{P}_H \dot{\mathbf{q}}_A,$$

which is equivalent to eq. (15) obtained in case of a single finger.

4.6.2 Case 2: motion of an object grasped in the hand with only the joints of the arm.

At the task level, we assume now that the same trajectory of the previous case is specified. It is now required that

$$\begin{bmatrix} \dot{\mathbf{x}}_f \\ \mathbf{0} \end{bmatrix} = \mathbf{J}_S \dot{\mathbf{q}}_S$$

with

$$\begin{aligned}\dot{\mathbf{q}}_H = \dot{\mathbf{q}}_{HA} + \dot{\mathbf{q}}_{HN} &= \mathbf{0}\\ \mathbf{J}_S\dot{\mathbf{q}}_N &= \mathbf{0}\end{aligned}$$

therefore

$$\begin{aligned}\mathbf{J}_{Pf}\dot{\mathbf{q}}_{PN} - \mathbf{J}_{Hf}\dot{\mathbf{q}}_{HA} &= 0\\ -\mathbf{J}_{Hd}\dot{\mathbf{q}}_{HA} &= 0.\end{aligned}$$

Hence, one computes

$$\dot{\mathbf{q}}_{PN} = \mathbf{J}_P^+\mathbf{J}_H\dot{\mathbf{q}}_{HA}$$

or

$$\dot{\mathbf{q}}_N = \begin{bmatrix} {}^B\mathbf{J}_P^+\,{}^B\mathbf{J}_H\dot{\mathbf{q}}_{HA} \\ -\dot{\mathbf{q}}_{HA}\end{bmatrix} = \begin{bmatrix} \mathbf{0} & {}^B\mathbf{J}_P^+\,{}^B\mathbf{J}_H \\ \mathbf{0} & -\mathbf{I}_f \end{bmatrix}\dot{\mathbf{q}}_A = \mathbf{P}_P\dot{\mathbf{q}}_A,$$

equivalent to eq. (18) obtained previously.

4.6.3 Case 3: deformation of the grasp triangle.

Let us consider now the internal motions only, which might be required to adjust the grasping forces. The desired trajectory in the task space implies that the task velocity vector is in the form:

$$\begin{bmatrix}\mathbf{0}\\ \dot{\mathbf{d}}\end{bmatrix} = \mathbf{J}_S\dot{\mathbf{q}}_S$$

i.e.

$$\begin{aligned}0 &= \mathbf{J}_{Pf}\dot{\mathbf{q}}_P + \mathbf{J}_{Hf}\dot{\mathbf{q}}_H\\ \dot{\mathbf{d}} &= \mathbf{J}_{Hd}\dot{\mathbf{q}}_H.\end{aligned}$$

It is clear that it is not possible to achieve this result while keeping the joints of the hand blocked ($\dot{\mathbf{q}}_H = 0 \Rightarrow \dot{\mathbf{d}} = 0$). In fact, the application of the projector $\mathbf{P}_P$ to the solution $\dot{\mathbf{q}}_A$ computed in this case by the algorithm, gives as result

$$\dot{\mathbf{q}}_S = \dot{\mathbf{q}}_A + \mathbf{P}_P\dot{\mathbf{q}}_A = \begin{bmatrix}\dot{\mathbf{q}}_{PA} + \mathbf{J}_P^+\mathbf{J}_H\dot{\mathbf{q}}_{HA}\\ \mathbf{0}\end{bmatrix}$$

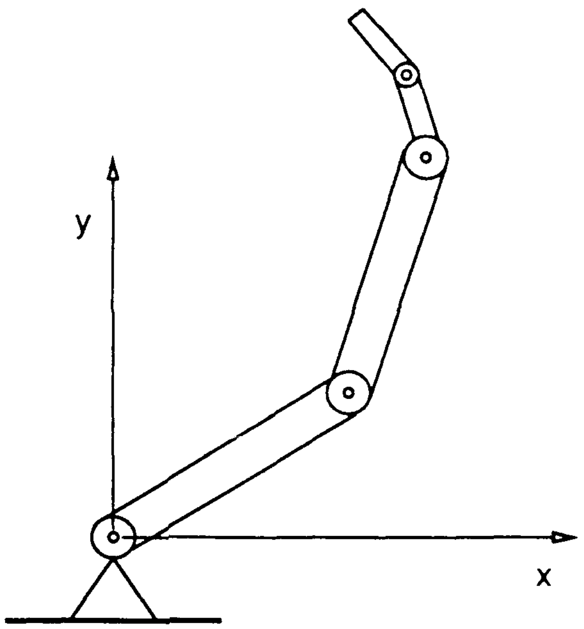


Figure 5: The simulated 4 degree-of-freedom redundant manipulator.

or, in terms of the final task space velocity,

$$\mathbf{J}_S\dot{\mathbf{q}}_S = \begin{bmatrix} \mathbf{J}_{Pf} & \mathbf{J}_{Hf} \\ 0 & \mathbf{J}_{Hd} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_{PA} + \mathbf{J}_P^+\mathbf{J}_H\dot{\mathbf{q}}_{HA} \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{Pf}\dot{\mathbf{q}}_{PA} + \mathbf{J}_{Pf}\mathbf{J}_P^+\mathbf{J}_H\dot{\mathbf{q}}_{HA} \\ 0 \end{bmatrix}$$

and therefore no internal motions are accomplished. Only the motion of the object which should be generated by $\dot{\mathbf{q}}_H$ are compensated for: nothing can be done for $\dot{\mathbf{d}}$. In this case, only the projector $\mathbf{P}_H$ could be applied without errors in the task space.

5 Simulation

In order to test the effectiveness of the proposed kinematic inversion technique, a simulation has been carried out with a planar redundant manipulator. The simulated robot is shown in Fig. 5. It has 4 degrees of freedom and it is basically constructed as two two-link manipulators emulating a planar two degrees of freedom arm carrying a two degrees of freedom finger. In Figs. 6-8 some results are reported.

In particular, in Figs. 6, 7 a trajectory of the end effector in the task space and the corresponding trajectories in the joint space, obtained from the algorithm of Fig. 1, are shown.

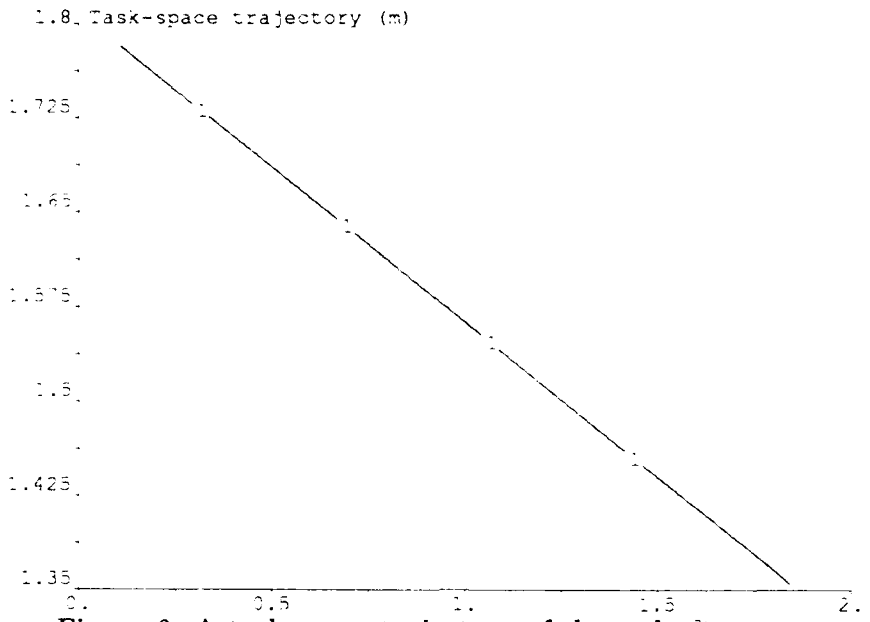


Figure 6: A task-space trajectory of the end-effector.

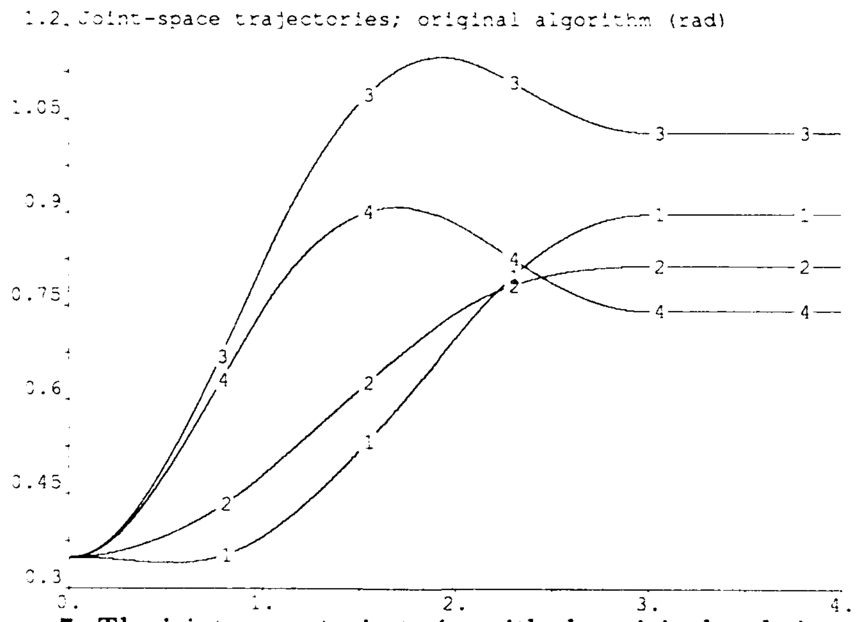


Figure 7: The joint-space trajectories with the original technique.

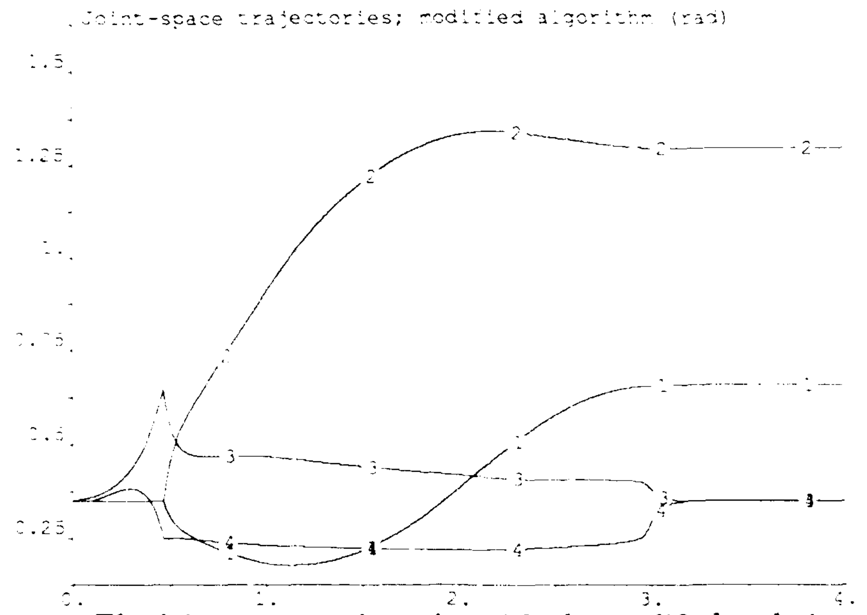


Figure 8: The joint-space trajectories with the modified technique.

Then, the same task has been performed using the algorithm described in the previous section. In Fig. 8 the obtained joint trajectories are shown. It may be noticed how only the joints 3, 4 are activated at the beginning of the task, while joints 1 and 2 start moving only when one of the condition in (20) fails. After that, joints 3 and 4 are restored to the original position, while only the first two joint actually move the end-effector along the cartesian trajectory. The tracking errors obtained with the original and the modified scheme are of the same order of magnitude (< 2 mm).

Figs. 9, 10, show a task in which the manipulator is required to apply a force. There are no motion specifications at the task level: only the requirement to exert a force along the negative x direction; a rigid surface is positioned at $x = 1.05$. Fig. 9 reports the joint position values. Again, joints 3 and 4 start the motion, and when one of them reaches the joint limit, the arm begins to move until contact with the surface is detected. Finally, during the force application phase, joints 3 and 4 are restored to the desired initial position. In Fig. 10 the desired and applied forces are shown.

6 Implementation

A first set of experiments of the above described technique has been carried out on an experimental set-up at the Artificial Intelligence Laboratory, M.I.T. The manipulator consists of a Puma 560 with the Salisbury Hand installed. The system has 15 degrees of freedom: 6 in the arm and 9 in the three fingers. Force information is available from force-sensors installed in the fingertips and from a sensorized palm. The control is performed by a two

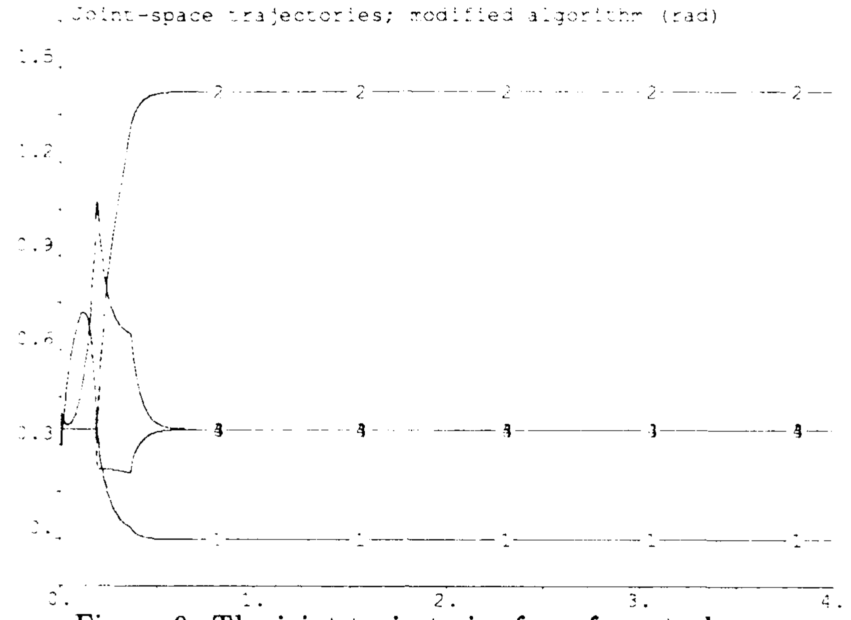


Figure 9: The joint trajectories for a force-task.

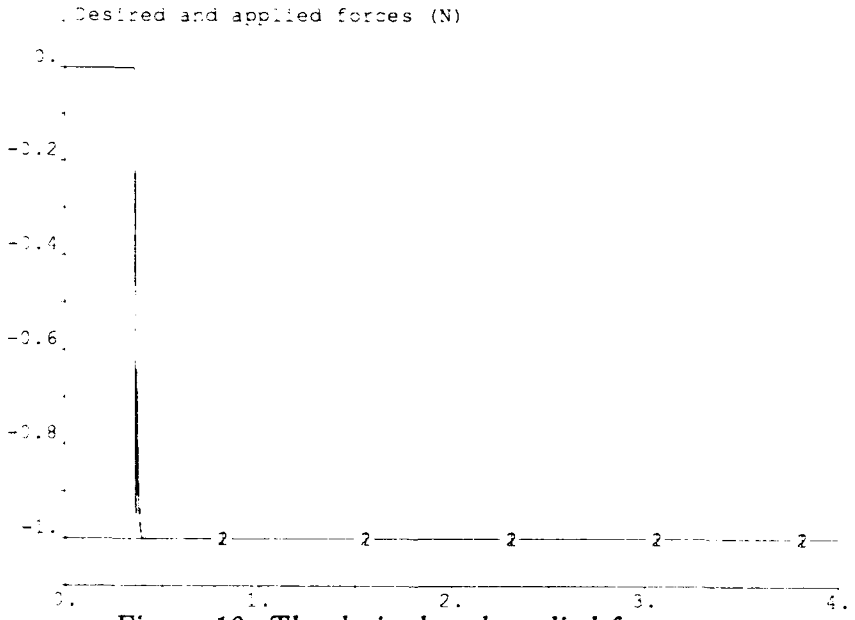
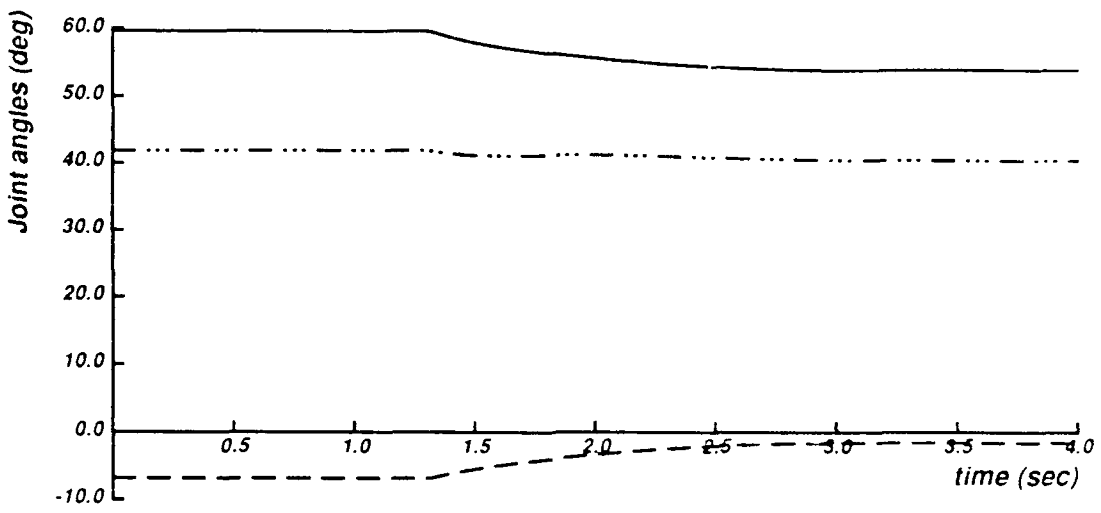


Figure 10: The desired and applied forces.



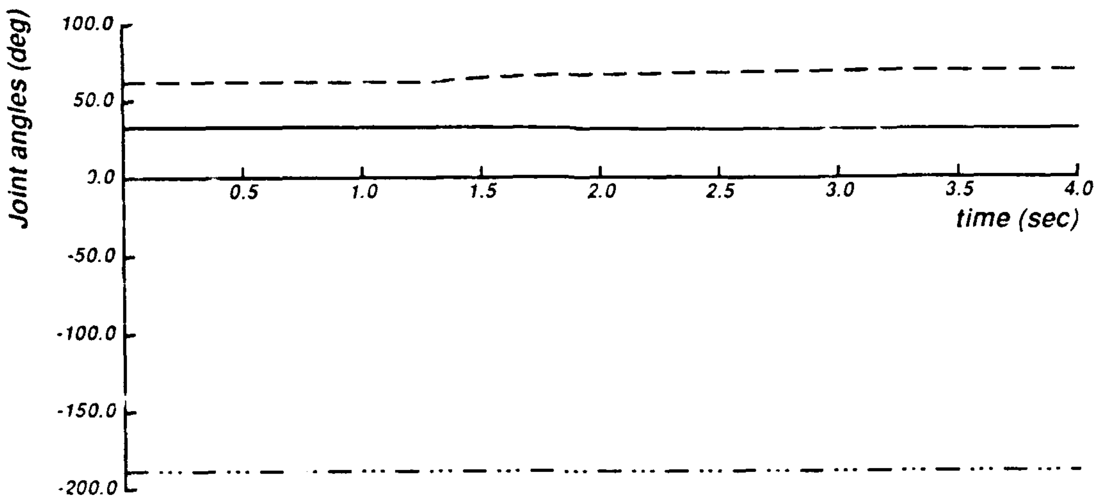
Trajectories of Puma joints 1-3

Figure 11: The trajectories of joints 1-3 of the Puma.

level control system: a VMEbus with three MC68030 processors running the VxWorks real time operating system is used as a supervisor for the servo level, based on three Unimation controllers, one for the Puma and two for the Hand. Two of the MC68030's are used to manage the communication with the controllers of the Puma and the Hand, as well as to acquire and process the force information from the fingertips. The third processor is used for the solution of the kinematic equations.

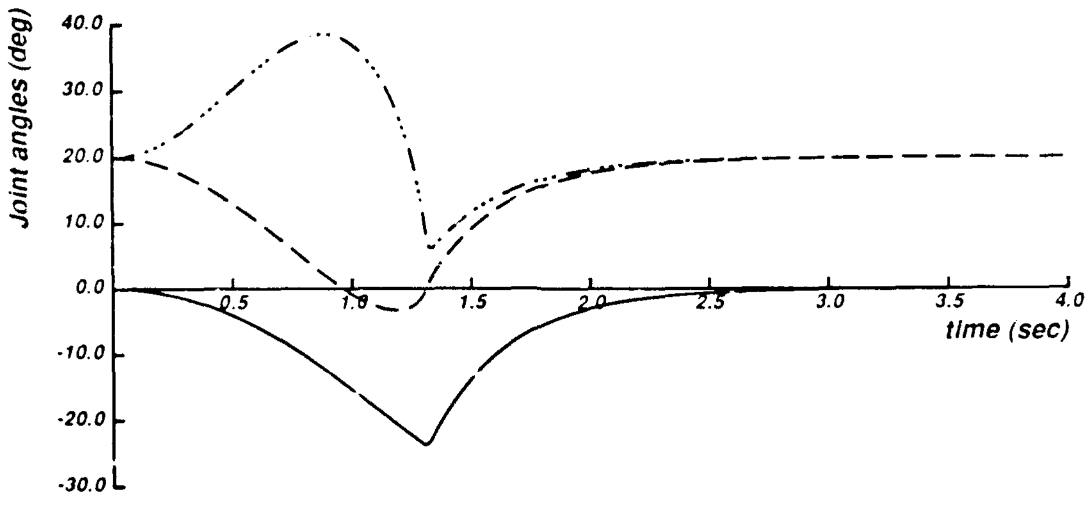
At the current stage of implementation, only one finger of the hand is taken into consideration in the kinematic inversion algorithm.

In Figs. 11-13, some results obtained from the experimental equivalent tasks of Figs. 6-10 are presented. Specifically, Figs. 11-12 show the Puma joint trajectories, and Fig. 13 the finger joint positions for a straight-line motion of the end effector. The modified algorithm is used in this case, leading to results similar to those presented in Fig. 8: the Puma's joints are blocked during the first period of motion, while in the last part the finger is restored to the initial position and the task is accomplished by the Puma.



Trajectories of Puma joints 4-6

Figure 12: The trajectories of joints 4-6 of the Puma.



Trajectories of finger joints

Figure 13: The trajectories of the finger's joints.

7 Conclusions and future work

In this report, work aimed at the development of a kinematic inversion algorithm for an hand-arm system has been described. The chosen approach for the determination of such algorithm has been to consider the device as a redundant manipulator, and to apply, with proper modifications, one of the most promising techniques in the field: the Jacobian transpose method. The modifications introduced in the scheme consider the different capabilities of the device in terms of maximum joint speed and/or amplitude of motion, as well as the possibility of executing the specified task with only a subset of the available joints. Moreover, a force feedback loop has been introduced, since the application of force on the environment is a major goal in the tasks for the system we consider.

At the present, only a partial implementation of the described algorithm, considering the arm and one finger, has been realized on an experimental set-up available at the Artificial Intelligence Laboratory, M.I.T., a Puma 560 with the Salisbury Hand.

The first comment on the currently realized algorithm concerns the rules adopted for the computation in real time of the gains α_i in (20)-(22). In fact, these rules take into consideration only static or first-order kinematic constraints, such as the joint-limits or the tracking errors. It could be of interest to take into consideration different and more general rules, based also on the effective dynamic capabilities of the individual joints.

Another interesting variation that could be introduced is to conceptually consider the wrist as a part of the hand rather than as a part of the arm. In this case, the "arm" would have only the responsibility to position the "hand" in the work-space, while all the remaining actions would be executed by the "hand". This should result in a more "anthropomorphic" behavior of the whole device, requiring no motion of the arm in manipulation tasks in which only small motions are required.

Finally, it is in the authors' opinion that the performances of the algorithm could be improved by considering the gain λ of the loop not simply as a scalar, but as a full $n_j x n_j$ matrix. As a matter of fact, when the forward kinematics function is not dimensionally homogeneous, some limitations of the performances, in terms of convergence of the algorithm to the solution, are noticed.

These modifications of the basic algorithm, along with the full implementation of the proposed technique on the hand-arm system, are among the main goals of the current activity.

References

[1] Salisbury, J.K., Craig, J.J., "Articulated Hands: Force Control and Kinematic Issues", Int. Journal of Robotic Research, Vol. 1, No. 1, Spring 1982.

[2] Jacobsen, S.C., Iversen, E.K., Knutti, D.F., Johnson, R.T., and Biggers, K.B., "Design of the UTAH/MIT Dexterous Hand", Proc. 1986 IEEE Int. Conf. on Robotics and Automation, San Francisco, CA, 1986.

[3] Bonivento, C., Caselli, S., Faldella, E., Laschi, R., Melchiorri, C., Tonielli, A., "Control System Design of a Dexterous Hand for Industrial Robots", IFAC SYROCO'88, Karlsruhe, RFG, 1988.

[4] B&L Engineering "The Belgrade/USC Robot Hand", Brochure, Preliminary Release, May 1989.

[5] Cutkosky, M.R. "Robotic Grasping and Fine Manipulation", Kluwer, Boston, MA, 1985.

[6] Iberall. T., "The Nature of Human Prehension: Three Dextrous Hands in One", Proc. IEEE Int. Conf. on Robotics and Automation, Raileigh, 1987.

[7] Kerr, J., Roth, B., "Analysis of Multifingered Hands", Int. Journal of Robotic Research, Vol. 4, No. 4, 1986.

[8] Pollard, N., Lozano-Perez, T., "Grasp Stability and Feasibility for an Arm with an Articulated Hand", Proc. IEEE Int. Conf. on Robotics and Automation, Cincinnati, OH, 1990.

[9] Roberts, K.S., "Coordinating a Robot Arm and Multi-Finger Hand Using the Quaternion Representation", Proc. IEEE Int. Conf. on Robotics and Automation, Cincinnati, OH, 1990.

[10] Lee, S., Lee, J.M., "Multiple Task Point Control of a Redundant Manipulator", Proc. IEEE Int. Conf. on Robotics and Automation, Cincinnati, OH, 1990.

[11] Sharon, A., Hogan, N., and Hardt, D.E., "High Bandwidth Force Regulation and Inertia Reduction Using a Macro/Micro Manipulator System", Proc. IEEE Int. Conf. on Robotics and Automation, Philadelphia, 1988.

[12] Egeland, O., "Task-Space Tracking with Redundant Manipulators", IEEE Journal of Robotics and Automation, Vol. RA-3, No. 5, Oct. 1987.

[13] Paul, R.P., Shimano, B., Mayer, G.E., "Kinematic Control Equations for Simple Manipulators", IEEE Trans. on System, Man and Cybern., Vol SMC-11, No. 6, June 1988.

[14] Pieper, D., "The Kinematics of Manipulators Under Computer Control", PhD. Thesis, Stanford Univ., Stanford, CA, 1986.

[15] Baker, D.R., Wampler, C.W., "On the Inverse Kinematics of Redundant Manipulators", Research Publication GMR-5478, Mathematics Dept., General Motor Research Laboratories, Warren, MI, July 1986.

[16] Nenchev, D.N., "Redundant Resolution Through Local Optimization: a Review", Jour. of Robotic Systems, 6(6), 1989.

[17] Ben-Israel, A., Greville, T.N.E., "Generalized Inverses: Theory and Applications", Wiley, 1974.

[18] Melchiorri, C., "Considerations About the Use of Minimum Norm Criteria for the Solution of Kinematic Problems", 1990 ACC, San Diego, CA, 1990.

[19] Griffis, M., Duffy, J., "Kinestatic Control: A Novel Theory for Simultaneously Regulating Force and Displacement", 21st. ASME Mech. Conf., Chicago, IL, 1990.

[20] Cleary, K., Tesar, D., "Incorporating Multiple Criteria in the Operation of Redundant Manipulators", Proc. IEEE Int. Conf. on Robotics and Automation, Cincinnati, OH, 1990.

[21] Baillieul, J., "Kinematic Programming Alternatives for Redundant Manipulators", Proc. IEEE Int. Conf. on Robotics and Automation, St. Luis, MO, 1985.

[22] Chang, P.H., "A Closed-Form Solution for the Control of Manipulators with Kinematic Redundancy", Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, CA, 1986.

[23] Maciejewski, A.A., Klein, C.A., "Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments", Int. Journal of Robotic Research, Vol. 4, No. 3, 1985.

[24] Nakamura, Y., Hanafusa, H., Yoshikawa, T., "Task-Priority Based Redundancy Control of Robot Manipulators", Int. Journal of Robotic Research, Vol. 6, No. 2, 1987.

[25] Balestrino, A., De Maria, G., Sciavicco, L., "Robust Control of Robotic Manipulators", Proc. 9th. IFAC World Congress, Budapest, Hungary, 1984.

[26] Wolovich, W.A., Elliott, H., "A Computational Technique for Inverse Kinematics", Proc. 23rd. Conf. on Decision and Control, Las Vegas, NV, Dec. 1984.

[27] De Maria, G., Marino, R., "A Discrete Algorithm for Solving the Inverse Kinematic Problem of Robotic Manipulators", Proc. 2nd. Int. Conf. on Advanced Robotics, Tokio, JIRA, 1985.

[28] Sciavicco, L., Siciliano, B., "Coordinate Transformation: A Solution Algorithm for One Class of Robots", IEEE Trans. Syst., Man, Cybern., Vol. SMC-16, No. 4, July/Aug. 1986.

[29] Sciavicco, L., Siciliano, B., "A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulator", IEEE Jour. of Robotics and Automation, Vol. 4, No. 4, Aug. 1988.

[30] Sciavicco, L., Siciliano, B., Chiacchio, P., "On the Use of Redundancy in Robot Kinematic Control", Proc. 1988 ACC, Madison, 1988.

[31] Balestrino, A., De Maria, G., Sciavicco, L., and Siciliano, B., "An Algorithmic Approach to Coordinate Transformation for Robotic Manipulators", Adv. Robotics, Vol. 2, 1988.

[32] Chiacchio, P., Chiaverini, S., Sciavicco, L., Siciliano, B., "Closed-Loop Inverse Kinematics Schemes for Constrained Redundant Manipulators with Task-Space Augmentation and Task-Priority Strategy", accepted for the publication on The Int. Jour. of Robotic Research, 1990.

[33] Slotine, J-J.E., Yoerger, D.R., "A Rule Based Inverse Kinematics Algorithm for Redundant Systems", Int. Jour. of Robotics and Automation, Vol. 2, No. 3, 1987.

[34] Das, H., Slotine, J-J.E., Sheridan, T.B., "Inverse Kinematic Algorithms for Redundant Systems", Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, CA, 1988.

[35] Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., "Numerical Recipes: The Art of Scientific Computing", Cambridge University Press, 1986.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
October 1990

3. REPORT TYPE AND DATES COVERED
memorandum

4. TITLE AND SUBTITLE
Exploiting the Redundancy of a Hand-Arm Robotics System

5. FUNDING NUMBERS
N00014-86-K-0685
NAG 319
CNR 89.00493.PF.67.3
NATO-CNR 215.23/15

6. AUTHOR(S)
Claudio Melchiorri and J. Kenneth Salisbury

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Artificial Intelligence Laboratory
545 Technology Square
Cambridge, Massachusetts 02139

8. PERFORMING ORGANIZATION REPORT NUMBER
AIM 1261

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Information Systems
Arlington, Virginia 22217

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
None

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution of this document is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In this report, a method for exploiting the redundant degrees of freedom of a hand-arm mechanical system for manipulation tasks is illustrated. The basic idea is to try to take advantage of the intrinsic capabilities of the arm and hand subsystems in terms of amplitude of motions, different velocity limits and degrees of precision for the achievement of a particular task. The Jacobian transpose technique, a well-known algorithm for the solution of the kinematic inverse problem, is at the core of the proposed method for the control of the hand-arm system. Different behaviors of the hand and of the arm are then obtained by means of constraints in $Null(\mathbf{J})$ added to the solution

(continued on back)

14. SUBJECT TERMS (key words)
redundant force control
control hand-arm

15. NUMBER OF PAGES
30

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

Block 13 continued:

given by the Jacobian transpose method. The constraints are generated by non-orthogonal projection matrices, computed on the basis of the behavior desired from the system, without resorting to extended task space techniques. Comments about the computation of the constraints, and how to take advantage of them, are reported in the paper, as well as a description of the experimental activity currently in progress on a robotic system (a Puma 560 with the Salisbury Hand) at the Artificial Intelligence Laboratory, M.I.T.